Evolutionary Topology in Truss Optimization via Dissolvable Beams

Ziwen Ye

CMU-CS-23-131
August 2023

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Lining Yao (Chair)
Jiaoyang Li

Submitted in partial fulfillment of the requirements for the degree of Master of Science.

Keywords: Truss Robot, Robotics, Multi-objective Optimization

Abstract

Variable Geometry Truss excels in shape-changing and reconfiguration, enabling it to perform a wide variety of motions, such as stretching, locomotion, and adapting to different tasks in various terrains. Despite all these advantages, most of these designs have a fixed topology. In this work, we aim to design a system of truss robots with variable topology by introducing the concept of dissolvable beams. Specifically, we leveraged a genetic algorithm-based computational pipeline to explore channel optimization and control policy before and after dissolving. We present the results of our system executing complex, coordinated tasks before dissolving and diverse, parallel tasks after dissolving.

Acknowledgments

I would like to thank Professor Lining Yao for being my advisors during my undergraduate and masters. I would like to thank my collaborator Tucker, my Mentor Jianzhe for the project. I would also like to thank Lea for helping with my thesis presentation. Last but not least, I would like to thank all my colleagues in morphing matter lab, my friends and family for their supports.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A long-standing challenge in robotics is designing a system that can perform a variety of tasks in different challenging environments. A group of robots named Variable Geometry Truss (VGT) offers a solution through shape-changing. These robots consist of length-changing beams following a mesh or truss-like structure. Applications involving a high degree of flexibility have been demonstrated, such as exploring planes [2, 5] or shoring up rubble at disaster sites [13].

In addition to those tasks, a diverse and challenging objective is to design a system in which both disconnected individual components and collectives can carry out distinct tasks. In nature, certain types of collectives exist wherein individual members of swarms are capable of executing simple tasks, yet they collectively perform complex tasks or move more efficiently [10]. For instance, groups of ants combine to form nests or bridges [8, 11], and slime molds aggregate to achieve collective locomotion [14].

Motivated by these inspirations, we aim to create a truss-based system capable of both individual and collective motions. Designing such a system typically involves altering the topology of the truss, as we need to break it down into sub-components. However, the majority of existing truss-based systems primarily focus on geometry changes, involving alterations to the length of the beams, and have scarcely delved into topology changes. In the case of studies that have explored topology change, like the Variable Topology Truss (VTT) [13], their primary emphasis lies in pushing the boundaries of reconfiguration beyond what can be achieved through geometry changes alone. The VTT also investigates topology change at a node level, and their system cannot be divided into disconnected components. Consequently, there is a necessity to introduce an innovative truss-based system that achieves topology changes by deconstructing the system into sub-components.

In this work, we propose a novel truss-based system that can break into subparts by introducing the ideas of dissolvable beams. Furthermore, we introduce an optimization pipeline based on genetic algorithms [16] for refining our geometry and topology configurations. Additionally, we showcase two applications of our system:

- **Collective Locomotion and Dispersion:** The system demonstrates the ability to move collectively and then disperse, traveling in different directions.
- **Grab, Locomote, and Recycle:** Our system can seize an object, move to a target location, and recycle one of its subcomponents back to the original position.

The primary challenge lies in optimizing topology configurations, which introduces an additional layer of complexity. This is because altering the connectivity can have a profound impact on the system's physics. For instance, disconnecting an edge can significantly alter the system's center of mass. As a result, optimizing locomotion for both the entire structure and its subparts becomes intricate, as global parameters can influence local parameters, and vice versa. The fundamental contribution of this research is the introduction of a dissolvable truss-based system along with an associated optimization pipeline.

In chapter 4, we will go through MetaTruss, a computational design framework for Pneumesh which I was involved in during my master's degree.

In chapter 5, we will go through the work of this thesis, how we incorporate ideas of dissolvable beams as an additional parameter in our optimization pipeline.

Chapter 2

Related Works

2.1 Truss Robots

A truss is a framework structure constructed from straight, rigid beams interconnected by joints. Its utilization is extensive in engineering and architecture due to its inherent structural efficiency. eferred to as Variable Geometry Truss (VGT) has embraced this architectural principle by substituting conventional fixed-length beams with beams of variable length. These variable-length beams are typically either electromechanical [12], employing electric motors for linear motion, or pneumatic [6], utilizing compressed air to facilitate movement. The alterations in the length of these beams lead to changes in the overall geometry of the structure. This innovative approach has found application across diverse domains, including robotic manipulators [7], locomotion systems [9], and adaptive morphing structures [17].

Despite their capacity for a significant degree of shape freedom, these robots are often constrained by rigid beams of limited length, tethered systems, and complicated control requirements. Certain work has addressed these problems. Usevitch [15] developed an untethered isoperimetric robot using pneumatic reel actuator to achieve high extension ratio. Pneumesh [4] used passive-stopper structure, and multi-way joint design to achieve multiple complex motion with a limited number of air channels.

All the Truss Robots mentioned above are capable of geometry changes, which involve altering their shapes by extending or contracting the lengths of the beams. However, none of them can modify their topology—referring to changes in the connections between joints and beams. The Variable Topology Truss (VTT) [13] introduced topology changes through a specialized joint structure known as the Spiral Zipper [1], enabling nodes to merge and split. This innovation allows the VTT to accomplish reconfiguration tasks that cannot be achieved solely through geometry changes.

In contrast to the Pneumesh and Isoperimetric robot systems, our approach focuses on investigating topology changes. Unlike the Variable Topology Truss (VTT), which explores topology changes through the introduction of the Spiral Zipper joint at the node level, we achieve topology modifications by incorporating dissolvable beams and altering graph connectivity at the edge level. The utilization of dissolvable components in our system further enables us to explore locomotion for shape-changing tasks in both the integral structure and its subparts. This exploration of locomotion within both scales represents an aspect that has not been previously explored by earlier truss-based robot designs.

Chapter 3

Background

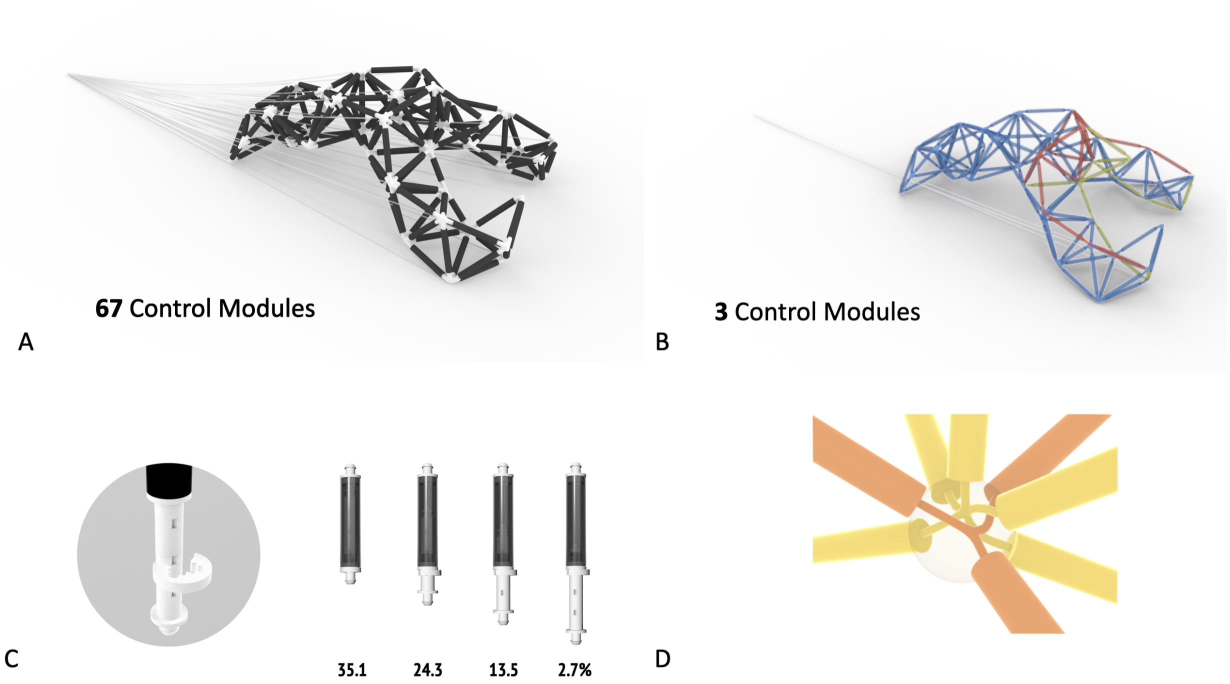

Figure 3.1: Overview of Pneumesh System [4]. A.number of controls needed without multi-way joint design. B.number of controls needed with multi-way joint design C. various beam lengths with blocker structure. D. multi-way joint design

3.1 Hardware System

We built our physical abstraction on top of existing Pneumesh framework 3.1. Pneumesh, a pneumatic truss-based shape-changing system, possesses the capability to perform multiple tasks using a minimal number of control units. Pneumesh is consisted of three key components: "adjustable pneumatic linear actuators, multi-way joints to separate air channels, and airflow valves controlling each channel" [4].

The Pneumesh framework employs a blocker structure to establish discrete lengths, effectively expanding the deformation potential without increasing the number of edges within the truss. Additionally, they utilize a "selectively-connected joint mechanism" that permits a subset of edges to be actuated simultaneously by a single actuator. This approach minimizes the quantity of control modules required while still enabling the truss to achieve diverse shapes through various combinations of open module statuses [4].

Their work introduces a design tool enabling users to customize truss structures for a range of locomotion and shape-changing applications. We opted for the Pneumesh platform as our physical system due to its simplicity of fabrication and ease of control. Notably, the Pneumesh system also incorporates passive beams, which can be conveniently replaced with dissolvable beams in our approach.

3.2 Optimization Algorithm

3.2.1 Genetic Algorithm

Genetic Algorithm [16] is a random search algorithm based on Darwin′s theory, and is mostly used in optimization for discrete systems. Genetic algorithm encodes a set of parameter to optimize as a gene. and follows the following pipeline

1. **Initialization**: create a gene Pool of size N_g filled with randomly generated genes

2. **Evaluation**: Evaluate the gene against a set of fitness functions
3. **Selection**: Select an elite pool of size N_l, where $N_l < N_g$ based on some selection schemes
4. **CrossOver**: CrossOver parameters between set of two genes to generate offsprings and add them to the gene pool
5. **Mutation**: Mutate the newly generated offsprings with low probability to add diversity into the population
6. **Replacement**: randomly fill in the rest of gene Pool and repeat the process of evaluation, selection, crossovers, and mutations.

Genetic algorithms offer several advantages, including their capability to tackle a diverse array of problems and exploit parallelism. However, they come with certain disadvantages, such as the need to fine-tune a multitude of parameters, including the formulation of the fitness function, population size, mutation and crossover rates, and selection criteria. Making an inappropriate choice in these aspects can render the algorithm difficult to converge and lead to outcomes lacking significance. A notable drawback lies in the selection criteria, which constitutes a key limitation.

3.2.2 NSGA-II

When applied to multi-objective optimization problems, traditional genetic algorithms often encounter challenges such as a deficiency in elitism, the high computational complexity of non-dominated sorting, and the necessity to define a sharing parameter to maintain diversity. NSGA-II [3] (Non-dominated Sorting Genetic Algorithm II) is considered as a fast and elitist multi-objective algorithm that effectively addresses these concerns. NSGA-II exhibits the following key features:

- **Elitism Preservation:** It preserves optimal solutions from earlier iterations to prevent their elimination.

- **Explicit Diversity Preservation:** The algorithm incorporates a distinct mechanism for preserving diversity, eliminating the requirement for a sharing parameter.
- **Reduced Computational Complexity:** NSGA-II introduces strategies that lead to a reduction in computational complexity, ensuring efficient performance.

These attributes collectively contribute to the strengths of NSGA-II, making it a powerful solution for multi-objective optimization tasks.

Chapter 4

MetaTruss

With Pneumesh, we possess a tool for forward simulation. Could we develop an algorithm to solve the inverse design problem? This problem involves creating a design that fulfills specific desired behaviors, such as achieving a target shape or locomotion pattern. In essence, given an initial model with position, topology, symmetry constraints, and defined objectives, is it possible to determine the relevant parameters for the physical system? These parameters include the contraction of each beam, channel grouping, and the action sequence for the truss.

To address this challenge, we built MetaTruss, a computational design pipeline. MetaTruss builds upon the physical foundation of Pneumesh and translates the physical attributes into a parameter set. It then uses genetic algorithm to find a set of optimal parameters for our multi-objective optimization problem.

4.1 Problem Statement

When we encode the physical system, we need to consider the truss model and its action sequences. We also need to consider representation of our abstract objective functions.

4.1.1 Model

In our physical system, we have a set of components, including joints, active beams as linear actuators, passive or dissolvable beams, and various air channels. We represent the truss model as a Graph G, consisted of vertices as V and edges as E, corresponding to beams and joints. Each edge $e \in E$ consists two vertices $(v_0, v_1), v_0, v_1 \in V$. In addition to this basic configuration, each edge contains an air channel $C \in n_c$ it belongs to, and an actuation length $l \in n_l$.

4.1.2 Action Sequences

We further parametrize the set of actuation Sequences. Each truss consists a series of actuation $A_0, A_1 \cdots A_n$ corresponding to the number of objective functions. Each actuation sequence is defined as a boolean matrix of size $n_a \times n_c$, where n_a is the number of action and n_c is the number of channels. 1 means contracting and 0 means inflated.

4.1.3 Objective Functions

Our mass-spring physical simulators take in a graph G and output a set of vertices v_s of size $(N, n_v, 3)$ N denotes the length of actuation, n_v denotes the number of edges in G, and 3 represents the (x, y, z) cartesian coordinates of each vertex. We also represent any specific vertex as either $v_{i,j,x}, v_{i,j,y}, v_{i,j,z}$ where i, j corresponds to the frame in the actuation sequence and the index of the vertex. We can then represent objective function regarding v_s. For example, if we want to represent an objective function of the truss moving towards the positive y direction, we can express it as

$$\textbf{max}[\sum(v_{n,j,y} - v_{0,j,y})]\forall j \in n_v$$

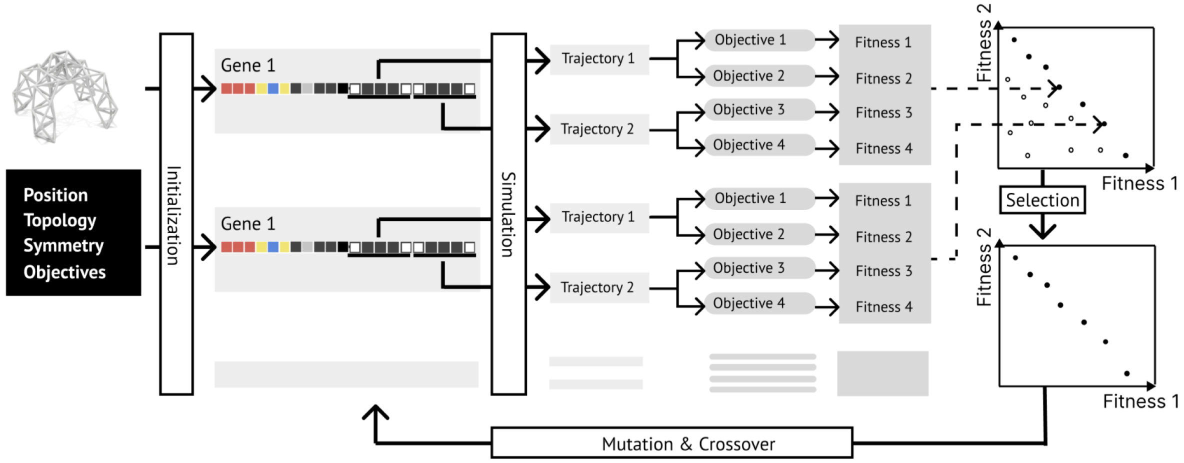

Figure 4.1: Overview of computational pipeline for MetaTruss

4.2 Computational Pipeline

4.2.1 Overview

We introduce a computational pipeline utilizing a combination of genetic algorithm and NSGA-II. This pipeline 4.1 incorporates tailored initialization, mutation, and crossover operators. The adoption of the genetic algorithm stems from the necessity to address the channel connection constraint through an optimization approach suitable for discrete data. The implementation of NSGA-II is motivated by the intention to optimize for multiple objectives.

4.2.2 Channel Constraint

When encoding parameter into gene, we need to taken into account the physical system. One specific constraint introduced by Pneumehs's multi-way joint system is channel connection. In the actual physical setup, we manage the flow of air through a common channel to control multiple joints using a single control signal. In the context of graph representation, it becomes crucial to guarantee that all connected edges belong to the same channel. Furthermore, within a given channel, every edge must be adjacent to at least one other edge through a joint in the same channel. This ensures the coherent functionality of the system while adhering to the physical constraints.

4.2.3 Genetic Operator

A gene comprises several elements: the positions of all nodes within the model, the topology, and action sequences. The positions and topology are predetermined. When a gene is being altered, we adjust its channel connectivity, edge contraction ratio, and action sequences. During the process of random assignment and modification of the gene, attention is given to maintaining adherence to the physical system's requirements, including channel constraints and the discrete set of contraction ratios.

- **initialization** During the initialization phase, we create a collection of random graph configurations, ensuring that each graph adheres to the channel constraint. Additionally, we generate a set of discrete edge contraction ratios and a boolean action sequence matrix through a random process.
- **Mutation**
- During the mutation phase, a random edge is selected, and an attempt is made to modify its channel by assigning it to one of its neighboring edges. It is crucial that the modified graph configuration maintains the connectivity within the same channel. An illustration of correct and incorrect mutations is depicted in 4.2. Additionally, the mutation process involves iterating through each action sequence and edge contraction ratio. This entails either flipping a bit or altering the value to a different contraction ratio, based on the probability of mutation.
- **CrossOver** Regarding crossover, it's important to note that we do not exchange channel configurations between two graphs. Instead, the crossover process involves iterating

through each action sequence and contraction ratio. Values are crossed over based on the assigned crossover probability.

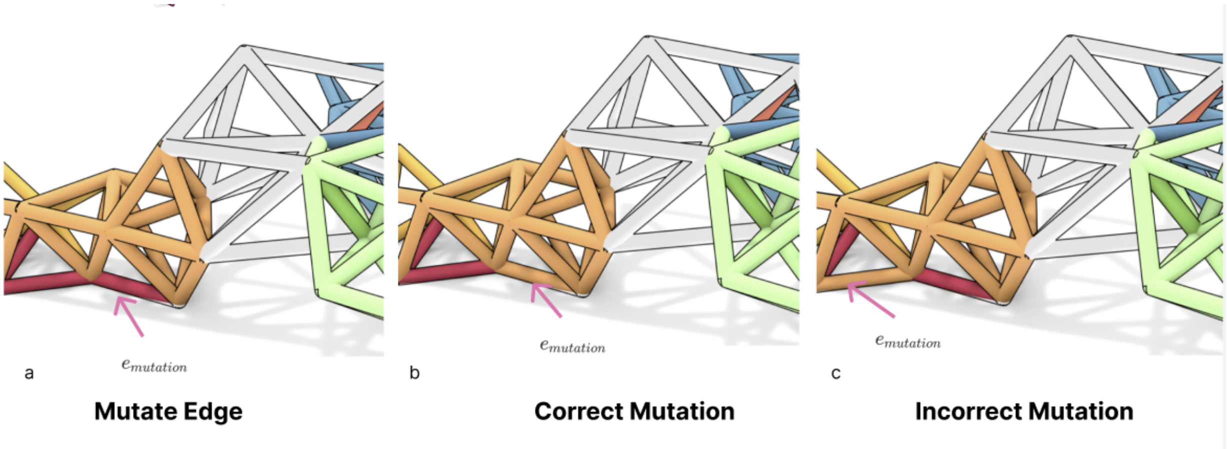

Figure 4.2: Illustration of edge mutation. a: initial edge setup. b: correct mutation c: incorrect mutation

4.2.4 Training Iterations

The training process maintains two distinct sets of gene pools, an active gene pool with capacity N_a, and an elite gene pool with size of N_e. Each iteration of the training contains n generations. During each generation, we follow the standard genetic algorithm pipeline with NSGA-II as evaluation algorithm to sort the genes based on their ranking and crowding distance. A fixed number of surviving genes are retained, while the rest are discarded. The survived genes undergo mutation and crossover, and new genes are randomly generated to replenish the pool. The surviving gene of the final generation in each iteration is placed into elite gene pool. When elite gene pool is full, all genes are reintegrated into the active gene pool. The use of an elite gene pool serves to mitigate the dominance of any single gene by temporarily removing elite genes from the active gene pool.

Chapter 5

Dissolvable Truss System

Extending beyond MetaTruss pipeline, we aim to explore an optimization pipeline that also addresses topology change in truss robots. In order to do so, we incorporate a set of new constraints, such as subgroups and dissolvable beams.

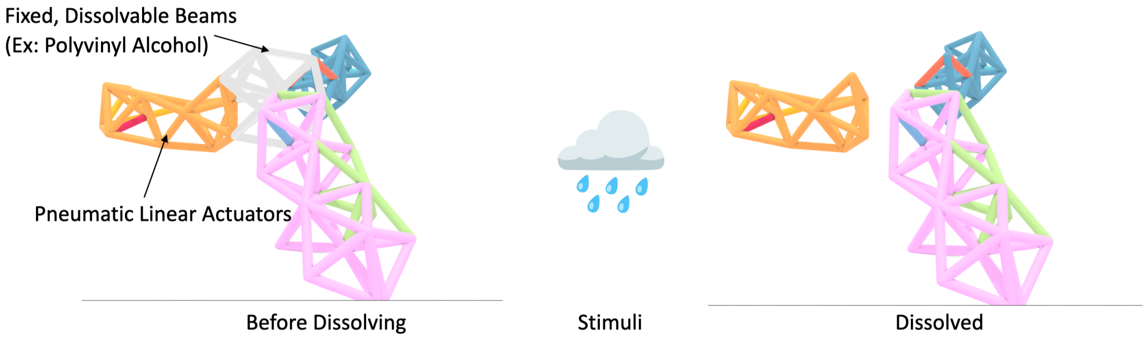

Figure 5.1: Demonstration of our dissolvable truss system. Passive beams break down with water stimuli

5.1 Dissolvable System

We propose a physical setup on top of Pneumesh, as shown in, 5.1, where a set of passive beams are replaced with dissolvable beams. In addition to the existing physical setup, we would incorporate passive beams with fixed lengths, fabricated from dissolvable materials. This is achievable due to the availability of materials that dissolve in response to specific stimuli. For instance, polyvinyl alcohol is a water-soluble synthetic polymer that is both easily manufacturable and possesses sufficient stiffness to support other beams. In certain imaginative scenarios, we envision a lobster looking robot, carrying an item and delivering it to a designated location. Subsequently, following a rain event, the robot disassembles into components, and recycles back the control modules on the tail component.

5.2 Group Constraint

The incorporation of dissolvable passive beams effectively partitions the graph into smaller subgraphs. edges within each subgraph G' can not formulate channel connection with edges outside of the group because in order to formulate disconnected individuals after dissolving, the edge must be adjacent to an edge within the group or a passive beam. This constraint ensures the integrity of the disassembled individuals after the dissolution process.

5.3 Problem Statement

Each edge is associated with a new parameter denoted a g, indicating the group to which that particular edge belongs. Additionally, dissolvable beams are encoded using a channel color of -1. The remainder of the process adheres to the established procedure.

5.4 Simulation

With the incorporation of objectives for both the integral truss system and the collective of disconnected subcomponents, we adapt by simulating the system in two modes: as a full graph denoted as G, and as a subgraph denoted as G^*, excluding the dissolvable beams. This approach allows us to effectively address the objectives for both scenarios.

5.5 Genetic Operators

During initialization, we simply grow channels for each subgraph following the same channel growing graph search algorithm. For the mutation step, a subgroup is randomly selected and a random edge within that subgroup is mutated following the normal mutation scheme. In crossover, we now introduce the ability to cross over graph configurations. Since each subgroup is disjoint, we can randomly select a subgroup and cross the subgraph configuration with ts counterpart from another gene. This enhanced approach to initialization, mutation, and crossover accounts for the new complexities introduced by the dual objectives and the use of dissolvable beams.

5.6 Adapted Computational Pipeline

In the overall pipeline, we incorporated a new parameter into the parameter space and adjusted the simulation accordingly we constrained the gene operators to execute initialization and mutation specifically within each subgroup. Additionally, a novel graph cross over operator was built to accomodate the disjoint nature of subgroups.

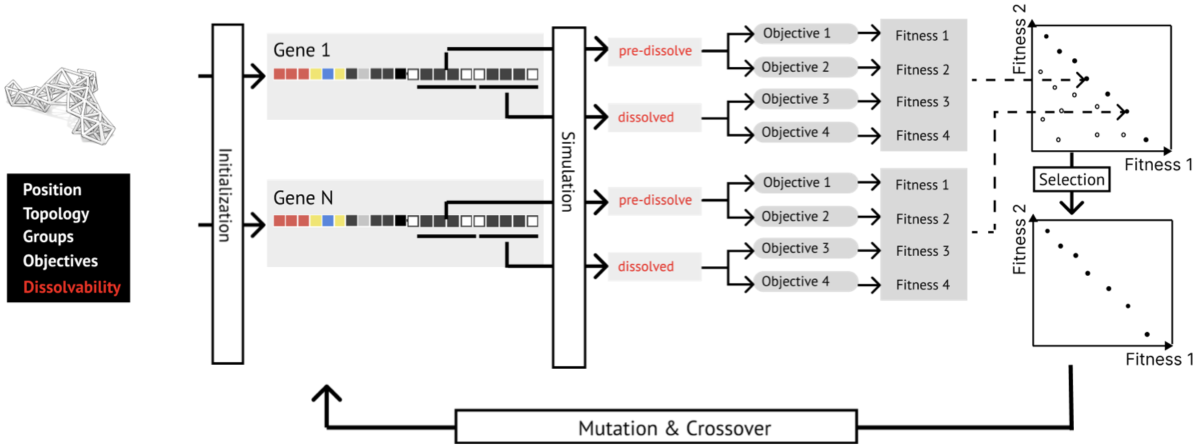

Figure 5.2: Modified computational pipeline by integrating dissolvable beams

Chapter 6

Results and Discussions

In our experiments with the dissolvable system, we utilize a lobster-looking truss robot with three subgroups and two channel for each subgroup 6.1. We optimize the parameter space for this robot to achieve two application tasks: 1. collective locomotion and dispersion 2. grab, locomote, and recycle. We then provide analysis and insights on the outcome of these optimizations.

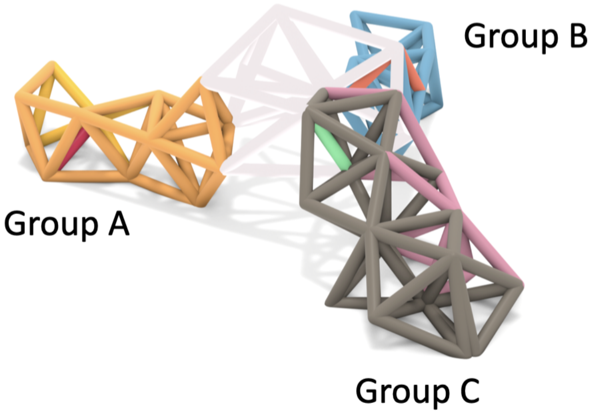

Figure 6.1: Dissolvable lobster-looking truss with three subgroups

6.1 Collective Locomotion and Dispersion

In this first task, we are thinking about a collective components doing locomotion while also having the ability to locomote as disconnected individuals. we aim to optimize three objective functions.

- The integral truss moving towards negative x direction.
- Subgroup A moves away from its origin.
- Subgroup B moves away from its origin.

Mathematically speaking, we can define the objectives correspondingly as

- **max**$[\sum(-(v_{n,j,x} - v_{0,j,x}))/n_v], j \in n_v$.
- **max**$[\sum(|(v_{n,j,x} - v_{0,j,x})| + |(v_{n,j,y} - v_{0,j,y})|)/n_A], j \in n_A$, where n_A denotes to the number of vertices in subgroup A.
- **max**$[\sum(|(v_{n,j,x} - v_{0,j,x})| + |(v_{n,j,y} - v_{0,j,y})|)/n_B], j \in n_B$, where n_B denotes to the number of vertices in subgroup B.

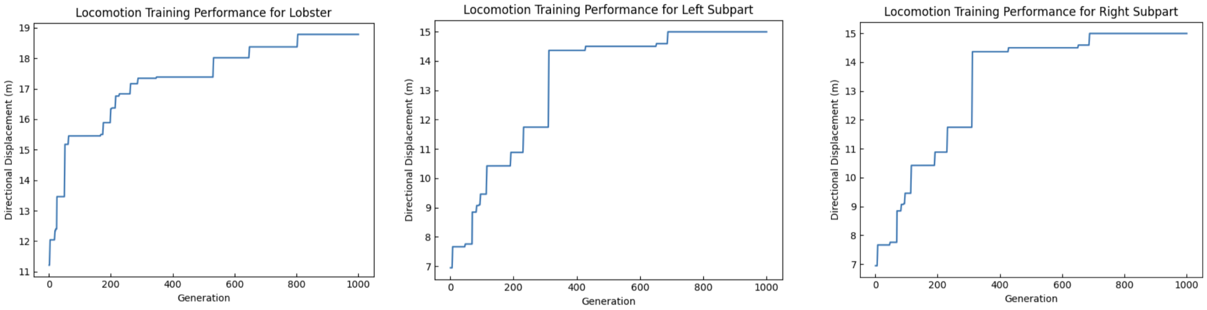

Figure 6.2: Locomotion performance over 1000 iterations

As we observed in the plot, the directional displacement for all three objective increases as training iteration increases, meaning that our optimization pipeline graudally converge to an optimal solution for each objective.

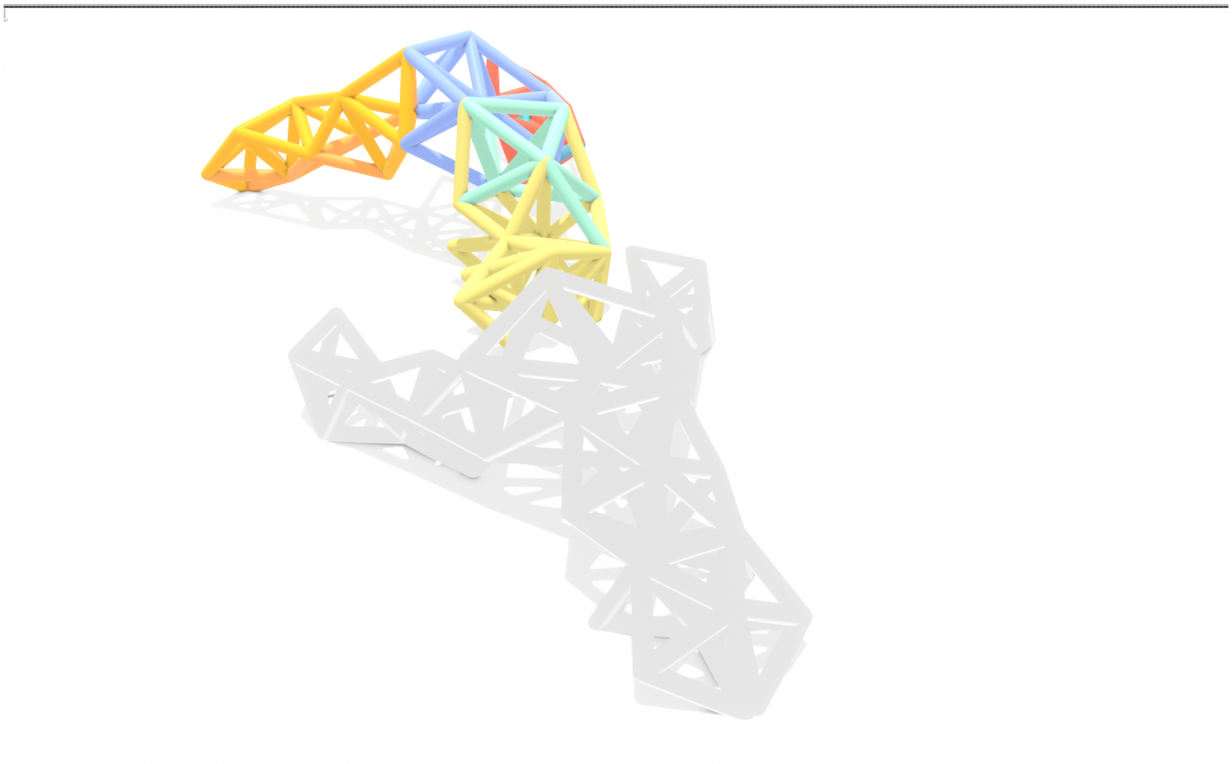

Figure 6.3: Displacement of lobster truss

6.2 Grab, Locomote, and Recycle

In the second scenario, we envision a truss robot conducting a field delivery task. This truss, resembling a lobster in appearance, is designed to grasp an object and transport it to a specified destination. Subsequently, the robot disassembles into three components, with the tail component and control modules being recycled. Although the subgroups remain the same in this case, the objective functions differ. We are optimizing three objective funcitons:

- The lobster truss grab and object.
- The lobster truss move towards negative x direction.
- Subgroup C move towards the direction from which the lobster originated.

Which can mathematically be represented as

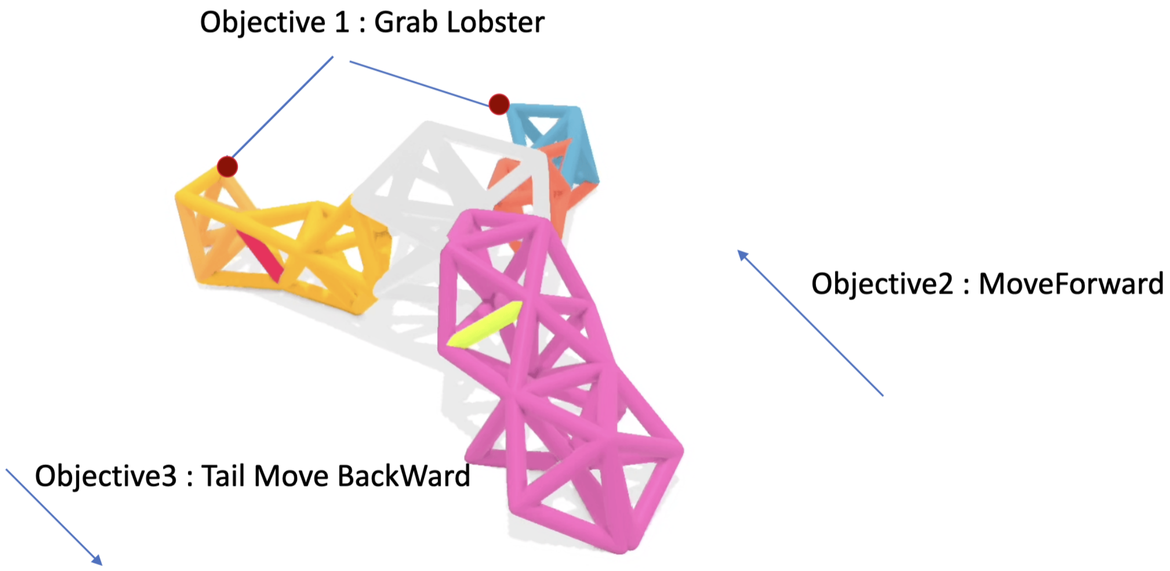

Figure 6.4: Schematic showing the model and objectives for Grab, locomote, and recycle task

- 6.4 Denoting the two highlighted vertices as l, k, we can represent grabbing an object as minimizing the max distance between v_l and v_k through the entire action sequence. $\mathbf{minmax}(v_{i,l} - v_{i,k}), i \in N$.
- $\mathbf{max}[-\sum((v_{n,j,x} - v_{0,j,x}))/n_v], j \in n_v$.
- $\mathbf{max}[\sum((v_{n,j,x} - v_{0,j,x}))/n_v], j \in n_C$, where n_C denotes the number of vertices in subgroup C.

6.5 We observe that, apart from the locomotion task, our system successfully optimizes for an alternative type of motion abstraction, specifically the action of ”grabbing”.

When evaluating the outcomes of our genePool through simulation visualization, we frequently observe instances of asymmetric design within our truss system. This asymmetry is characterized by non-mirrored channel connections and contraction ratios along specific axes.

The truss robot often encounters challenges when attempting to move directly to a target lo-

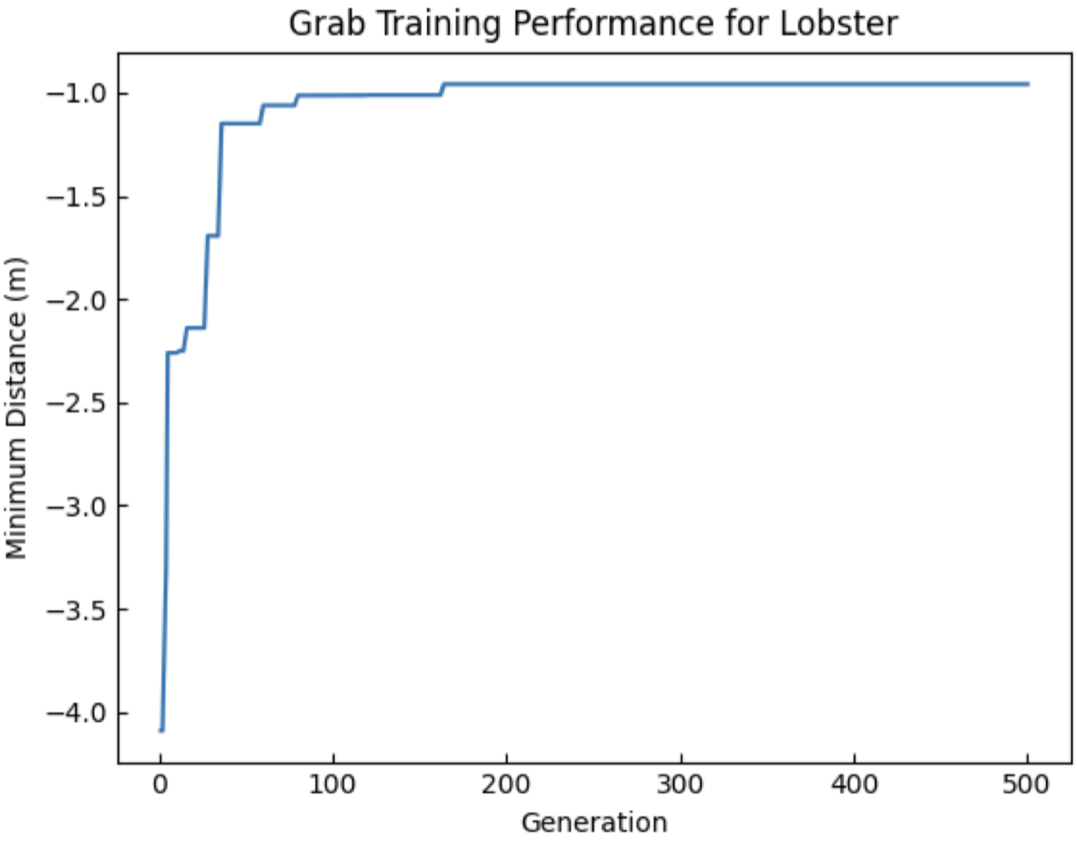

Figure 6.5: Minimum distance between two vertices over 500 iterations

cation, exhibiting a tendency to follow a circular trajectory instead 6.6. However, previous work in metaTruss has demonstrated that the introduction of a symmetry constraint yields significant benefits. By enforcing such a constraint, we can ensure symmetrical movement patterns and simultaneously reduce the search space. This enhancement expedites the computation pipeline and enhances the overall efficiency of our approach.

It is important to note that incorporating a symmetry constraint alongside the existing group constraints introduces an additional layer of complexity. However, it can a potential direction for future explorations and enhancements in our truss robot design and optimization process.

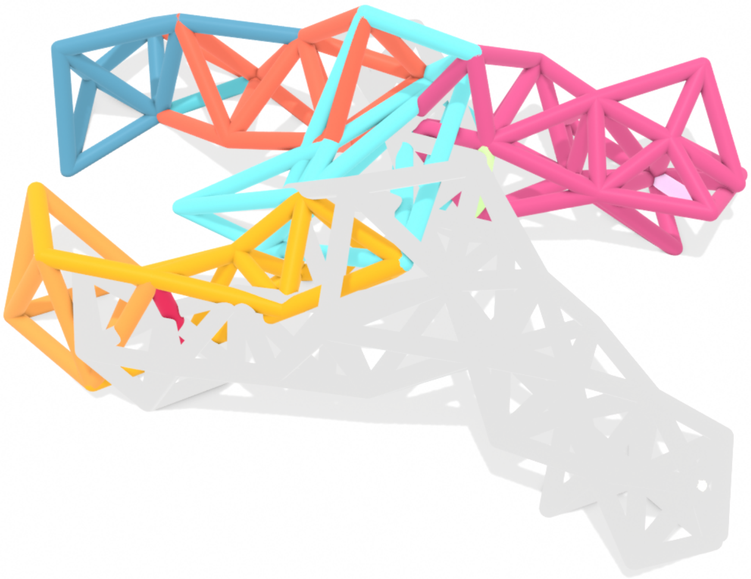

Figure 6.6: Displacement of lobster truss when performing grabbing and locomote

Chapter 7

Future Works

We have showcases preliminary findings from our proposed system, and we are confident that a wide array of tasks can be explored within the framework of our dissolvable truss-based system.

First, we would like to delve into shape-changing applications, broadening the scope and diversity for our objective functions. This exploration can encompass two distinct approaches:

- **Shape-changing through actuation:** Investigating the feasibility of achieving shape changes in the truss structure through controlled actuation mechanisms.
- **Shape-changing through strong buckling force induced by rigid structure:** Exploring the concept of inducing significant buckling forces within a rigid truss structure by selectively removing specific beams.

In addition, we would like to modify the computation pipeline to optimize for our specific system. To achieve this, we propose a novel approach. Initially, we intend to train the gene pool for a specified number of iterations, leading to the identification of a set of relatively high-quality genes located within the pareto fronts. Subsequently, we will implement a distinct modification to our pipeline. This involves the random selection of a gene from this set and the execution of our pipeline using the same model configuration, but different action sequences. This modification is unique to our system because, once the subcomponents become disconnected entities, the action sequences of each sub-objective do not interfere. Therefore, it is possible for the system to converge faster, as we can cross over a wide range of actions while constraining the model configuration to a locally optimal choice.

Last but not least, we would like to explore incorporation of a set of dissolvable beams as an optimization parameter. Our existing setup operates under the assumption of a predetermined set of passive beams and predefined groups. However, we aim to extend our approach by introducing the concept of dissolvable beams as a dynamic group. By treating dissolvable beams as a variable within our gene operators, we have the potential to uncover solutions that align more precisely with the specific objectives we seek to attain.

Bibliography

[1] Foster Collins and Mark Yim. Design of a spherical robot arm with the spiral zipper prismatic joint. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 2137–2143. IEEE, 2016. 2.1

[2] Steven Curtis, Matthew Brandt, Greg Bowers, Gary Brown, Cynthia Cheung, Caner Cooperider, Mike Desch, Noah Desch, John Dorband, Kyle Gregory, et al. Tetrahedral robotics for space exploration. In *2007 IEEE Aerospace Conference*, pages 1–9. IEEE, 2007. 1

[3] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. doi: 10.1109/4235.996017. 3.2.2

[4] Jianzhe Gu, Yuyu Lin, Qiang Cui, Xiaoqian Li, Jiaji Li, Lingyun Sun, Cheng Yao, Fangtian Ying, Guanyun Wang, and Lining Yao. Pneumesh: Pneumatic-driven truss-based shape changing system. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391573. doi: 10.1145/3491102.3502099. URL `https://doi.org/10.1145/3491102.3502099`. (document), 2.1, 3.1, 3.1

[5] Gregory J Hamlin and Arthur C Sanderson. Tetrobot modular robotics: Prototype and experiments. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS'96*, volume 2, pages 390–395. IEEE, 1996. 1

[6] Zachary M. Hammond, Nathan S. Usevitch, Elliot W. Hawkes, and Sean Follmer. Pneumatic reel actuator: Design, modeling, and implementation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 626–633, 2017. doi: 10.1109/ICRA.2017.7989078. 2.1

[7] Peter C Hughes, Wayne G Sincarsin, and Kieran A Carroll. Trussarm—a variable-geometry-truss manipulator. *Journal of Intelligent Material Systems and Structures*, 2(2): 148–160, 1991. 2.1

[8] Melinda Malley, Bahar Haghighat, Lucie Houel, and Radhika Nagpal. Eciton robotica: Design and algorithms for an adaptive self-assembling soft robot collective. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4565–4571. IEEE, 2020. 1

[9] Koryo Miura, Hiroshi Furuya, and Kenichi Suzuki. Variable geometry truss and its application to deployable truss and space crane arm. *Acta astronautica*, 12(7-8):599–607, 1985. 2.1

[10] Yogeswaran Mohan and SG Ponnambalam. An extensive review of research in swarm robotics. In *2009 world congress on nature & biologically inspired computing (nabic)*, pages 140–145. IEEE, 2009. 1

[11] Francesco Mondada, Luca Maria Gambardella, Dario Floreano, Stefano Nolfi, J-L Deneuborg, and Marco Dorigo. The cooperation of swarm-bots: Physical interactions in collective robotics. *IEEE Robotics & Automation Magazine*, 12(2):21–28, 2005. 1

[12] Eleftheria S Sergaki, George S Stavrakakis, and Anastasios D Pouliezos. Optimal robot speed trajectory by minimization of the actuator motor electromechanical losses. *Journal of Intelligent and Robotic Systems*, 33:187–207, 2002. 2.1

[13] Alexander Spinos and Mark Yim. Towards a variable topology truss for shoring. In *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*,

pages 244–249. IEEE, 2017. 1, 2.1

[14] Takuya Umedachi, Koichi Takeda, Toshiyuki Nakagaki, Ryo Kobayashi, and Akio Ishiguro. Fully decentralized control of a soft-bodied robot inspired by true slime mold. *Biological cybernetics*, 102:261–269, 2010. 1

[15] Nathan S. Usevitch, Zachary M. Hammond, Mac Schwager, Allison M. Okamura, Elliot W. Hawkes, and Sean Follmer. An untethered isoperimetric soft robot. *Science Robotics*, 5(40): eaaz0492, 2020. doi: 10.1126/scirobotics.aaz0492. URL `https://www.science.org/doi/abs/10.1126/scirobotics.aaz0492`. 2.1

[16] Xin-She Yang. Chapter 6 - genetic algorithms. In Xin-She Yang, editor, *Nature-Inspired Optimization Algorithms (Second Edition)*, pages 91–100. Academic Press, second edition edition, 2021. ISBN 978-0-12-821986-7. doi: https://doi.org/10.1016/B978-0-12-821986-7.00013-5. URL `https://www.sciencedirect.com/science/article/pii/B9780128219867000135`. 1, 3.2.1

[17] Chih-Han Yu, Kristina Haller, Donald Ingber, and Radhika Nagpal. Morpho: A self-deformable modular robot inspired by cellular structure. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3571–3578. IEEE, 2008. 2.1